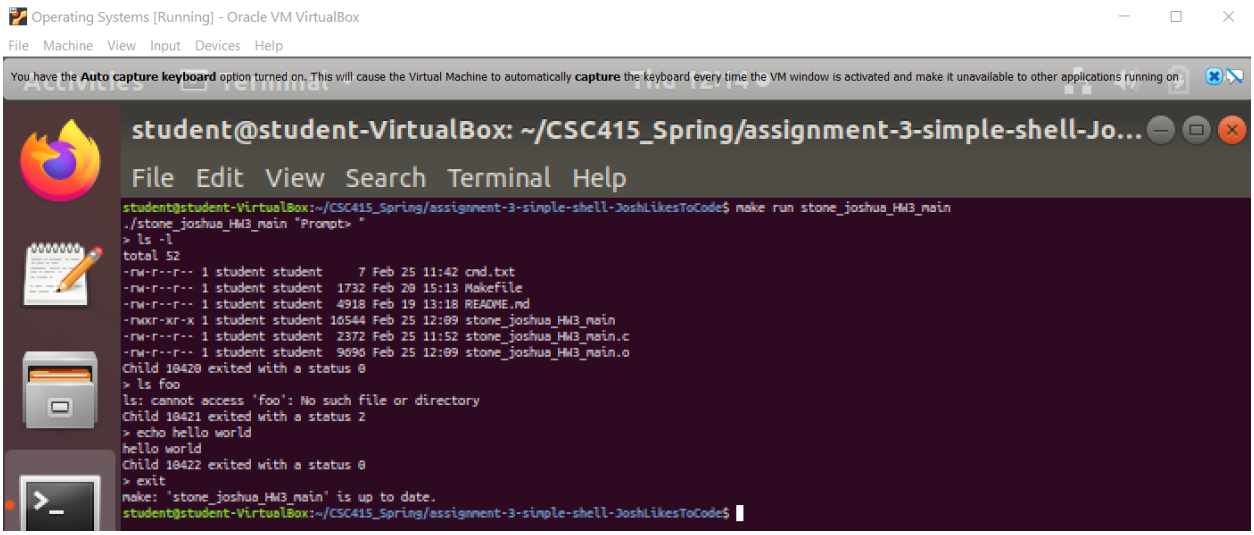
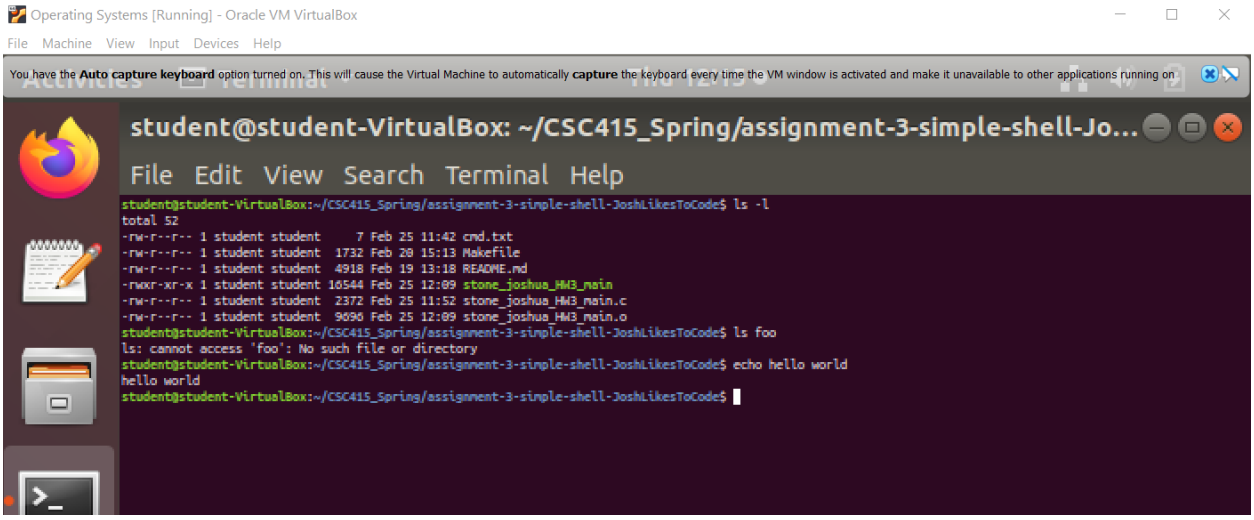
For this program I created a simple shell that parses through user input and then splits that input into tokens which will later be used as commands for our simple shell. Some of the problems I ran into with this assignment included having execvp() run infinitely, which I solved by making the program exit when the user_input was a blank space or a newline character. Another major problem I had to overcome when programming this assignment was being able to read commands from a .txt file. The big issue I had with this problem is after the .txt file was parsed and the execvp() executed it's command, then my program would throw a segmentation fault and exit. I solved this problem with the same logic that solved execvp() running infinitely – when user_input is a blank space or a newline (or EOF) the program will exit gracefully, but before that it will parse the .txt file's command and execute it. The last issue I ran into was related to exiting, where after the user typed "exit" when prompted, the program would display a final "Child %d exited with a status %d" and then exit, which was not the desired way to exit the program. I solved this by moving the function that checks if user_input == "exit" above the execvp() function, so that way the program exits before it executes any further commands. I have included screen shots below:

Here's the screenshot of the compilation along with the simple shell:



Here's the screen shot of the same commands used in the command line (for validation) :




Lastly, here's the screen shot of the compilation when ran with < cmd.txt which contains the same commands as the above examples within the .txt file: